

1                   **AUTOMATIC STATUS POLLING FAILOVER OF DEVICES IN A**  
2                   **DISTRIBUTED NETWORK MANAGEMENT HIERARCHY**  
3  
4

5                   **CROSS REFERENCES TO RELATED APPLICATIONS**

6                   The subject matter of the present application is related to copending United  
7                   States application, Serial No. 08/705,358, titled "Distributed Internet Monitoring  
8                   System and Method", Docket No. 10950961-1, filed August 29, 1996; copending  
9                   United States application, Serial No. 08/947,219, titled "Network Management Event  
10                  Correlation in Environments Containing Inoperative Network Elements", Docket No.  
11                  10971522-1, filed October 8, 1997; and copending United States application, Serial  
12                  No. 08/551,499, titled "Filtering System and Method for High Performance Network  
13                  Management MAP, Docket No. 10950101-1, filed November 1, 1995, all of which  
14                  are assigned to the assignee hereof and are herein incorporated by reference.

15  
16                  **FIELD OF THE INVENTION**

17  
18                  The present invention relates generally to data communications  
19                  networks and, more particularly, to a system and a method for automatic  
20                  status polling failover of devices in a distributed data communications network.  
21

22                  **BACKGROUND OF THE INVENTION**

23                  A data communications network generally includes a group of devices,  
24                  or objects, such as computers, repeaters, bridges, routers, etc., situated at  
25                  network nodes and a collection of communication channels or interfaces for  
26                  interconnecting the various nodes. Hardware and software associated with

1 the network and the object devices on the network permit the devices to  
2 exchange data electronically via the communication channels.

3  
4 The size of a data communications network can vary greatly. A local  
5 area network, or LAN, is a network of devices in close proximity, typically less  
6 than a mile, that are usually connected by a single cable, such as a coaxial  
7 cable. A wide area network (WAN) is a network of devices separated by  
8 longer distances and often connected by telephone lines or satellite links, for  
9 example. Some WANs span the United States, as well as the world.  
10 Furthermore, many of these networks are widely available for use by the  
11 public, including universities and commercial industries.

12  
13 A very popular industry standard protocol for data communication in  
14 networks is the Internet Protocol (IP). This protocol was originally developed  
15 by the U.S. Department of Defense, and has been dedicated to public use by  
16 the U.S. government. In time, the Transmission Control Protocol (TCP) and  
17 the Unreliable Datagram Protocol (UDP) were developed for use with the IP.  
18 The TCP/IP protocol is a protocol that implements certain check functionality  
19 and thus guarantees transfer of data without errors. The UDP/IP protocol  
20 does not guarantee transfer of data but it offers the advantage of requiring  
21 much less overhead than does the TCP/IP protocol. Moreover, in order to  
22 keep track of and manage the various devices situated on a network, the  
23 Simple Network Management Protocol (SNMP) was eventually developed for

1 use with the UDP/IP platform. The use of these protocols has become  
2 extensive in the industry, and numerous vendors now manufacture many types  
3 of network devices capable of operating with these protocols.  
4

5 Network Management Systems, such as OpenView Network Node  
6 Manager (NNM) by Hewlett-Packard Company of Palo Alto, California, are  
7 designed to discover network topology (i.e., a list of all network devices or  
8 objects in a domain, their type, and their connections), monitor the health of  
9 each network object, and report problems to the network administration (NA).  
10 NNM contains a monitor program called *netmon* that monitors the network;  
11 NNM is capable of supporting a single *netmon* program in the case of a non-  
12 distributed network management environment and multiple *netmon* programs  
13 in the case of a distributed network management environment. In the  
14 distributed network management environment, a plurality of netmon processes  
15 run on various **Collection Station** hosts, each of which communicates  
16 topology and status information to a centralized control unit, called a  
17 **Management Station**, that presents information to the NA. The management  
18 station is configured to discover the network topology and from that, construct  
19 a network management map comprised of various submaps typically arranged  
20 in a hierarchical fashion. Each submap provides a different view of the  
21 network and can be viewed on a display device.  
22

1 The monitoring function of a Network Management System is usually  
2 performed by a computer program that periodically polls each network object  
3 and gathers data that is indicative of the object's health. Thus, each collection  
4 station is responsible for polling of objects assigned to it while the  
5 management station is assigned to poll objects assigned to it. Based upon the  
6 results of the poll, a status value will be determined. For example, a system  
7 that fails to respond would be marked as "critical." *netmon* performs the status  
8 polling function.

9  
10 It is important to the proper operation of the network that the failure of  
11 any network object be known as soon as possible. The failure of a single  
12 network object can result in thousands of nodes and interfaces suddenly  
13 becoming inaccessible. Such a failure must be detected and remedied as  
14 soon as possible. Since collection stations are responsible for detecting the  
15 failure of their network objects through status polling, when a collection station  
16 itself goes down alternate arrangements must be made to ensure that status  
17 polling of the failed objects is maintained.

18  
19 When a collection station has been downgraded from a normal status to  
20 a critical status due to an inability to communicate with the collection station,  
21 the objects normally polled by the critical collection station must continue to be  
22 polled. One way to ensure that a collection station's object are properly polled  
23 on a periodic basis is to build in redundancy to the network management

1 system. A set of objects are thus polled by the management station as well as  
2 by the collection station. This practice of redundancy, however, while  
3 operating to ensure polling of objects has the disadvantage of increasing  
4 overhead costs of the network. Having a set of objects polled by both its  
5 collection station and the management station is, of course, inefficient for the  
6 vast majority of time during which such redundant polling is not necessary.  
7 There is therefore an unmet need in the art to be able to ensure that objects of  
8 a collection station will be status polled in a non-redundant manner in the  
9 event that the collection station is downgraded from a normal to a critical  
status.

## SUMMARY OF THE INVENTION

It is therefore an object of the present invention to ensure that objects of a collection station will be status polled in a non-redundant manner in the event that the collection station is downgraded to a critical status.

Therefore, according to the present invention, an automatic failover methodology is provided in which a central control unit will automatically takeover status polling for a collection station that is or becomes temporarily unreachable. The automatic failover feature of the present invention is accomplished by a network monitor program that resides on the central control unit. The network monitor program operates to quickly take over status polling for network objects that are managed by a collection station that has been downgraded to a critical status. When the collection station has returned to normal status, the network monitor program will stop status polling objects for the collection station and the collection station will again resume status polling of the objects. The present invention is applicable to any distributed computing environment, such as a data communications network, in which it is desirable to have a central control unit assume the interface status polling operation of a temporarily inaccessible collection station.

## BRIEF DESCRIPTION OF THE DRAWINGS

The novel features believed characteristic of the invention are set forth in the claims. The invention itself, however, as well as the preferred mode of use, and further objects and advantages thereof, will best be understood by reference to the following detailed description of an illustrative embodiment when read in conjunction with the accompanying drawing(s), wherein:

**Figure 1a** illustrates a network topology of a distributed computing environment before the failover mechanism of the present invention has occurred;

**Figure 1b** illustrates a collection station that has become unreachable and therefore downgraded from a normal to critical status;

**Figure 2** illustrates an overview of the methodology of the present invention;

**Figure 3** illustrates the methodology of the present invention used to determine whether a collection station has become unreachable;

**Figure 4** illustrates the methodology of the present invention for loading the topology of a critical collection station;

1  
2           **Figure 5** illustrates a pointer to the topological information for a  
3 particular collection station, according to the present invention;  
4

5           **Figure 6** illustrates that the topologies for one or more critical collection  
6 stations may be loaded onto the central control unit, according to the present  
7 invention;  
8

9           **Figure 7** illustrates the methodology for releasing the topology  
10 associated with a collection station that has become accessible again,  
11 according to the present invention; and  
12

13           **Figure 8** is a state diagram that illustrates the operation of the present  
14 invention.  
15



1       **DESCRIPTION OF THE INVENTION**

2               The automatic failover methodology of the present invention provides a  
3       mechanism whereby a central control unit, such as a management station, will  
4       automatically takeover interface status polling for a collection station that is  
5       temporarily unreachable. The present invention is applicable to any distributed  
6       computing environment, such as a data communications network, in which it is  
7       desirable to have a central control unit assume the interface status polling  
8       operation of a temporarily inaccessible collection station. The collection  
9       station may be inaccessible due to the network on which the central control  
10      unit and the collection station reside being down or the collection station being  
11      down for maintenance.

12  
13              The automatic failover feature of the present invention is accomplished  
14      by a network monitor program, called *netmon*, that resides on the central  
15      control unit. *Netmon* operates to quickly take over status polling for network  
16      interfaces managed by a collection station that has been downgraded to a  
17      critical status. When the collection station has returned to normal status,  
18      *netmon* will stop status polling interfaces for the collection station.

19  
20              Upon a collection station becoming temporarily unreachable, *Netmon* on  
21      the central control unit receives a list of network nodes managed by the  
22      collection station, which can be restricted by a failover filter. The default action  
23      is to supply *netmon* with the full loaded topology for the collection station that

1 has gone down, thereby allowing the central control unit to take over entirely  
2 for the critical collection station. Duplicate objects shared by both the central  
3 control unit and the collection station will only be polled once from the central  
4 control unit. If, however, multiple collection stations are polling the nodes and  
5 one of the collection stations is downgraded to critical status, then both the  
6 central control unit and the remaining operational collection stations will poll  
7 the node. The central control unit performs this duplicate polling for the critical  
8 collection station because it is desirable to maintain the polling configuration  
9 defined by the user.

10  
11 Referring to **Figure 1a**, an example of a network topology before the  
12 failover mechanism of the present invention has occurred is shown. In this  
13 representation, it can be seen that the collection station CS 12 polls a  
14 collection of objects 16 through communication interface 22 while central  
15 control unit, shown here as management station MS 14, polls a collection of  
16 objects 18 through communication interface 24. As shown in the figure, in this  
17 example there is some commonality, or overlap, between collection of objects  
18 16 and 18; the present invention operates regardless of the existence of such  
19 an overlap. Collection station CS 12 and MS 14 communicate with each other  
20 over communication interface 20, as shown.

21  
22 Referring now to **Figure 1b**, the collection station CS 12 has become  
23 unreachable for some reason and therefore been downgraded from normal

1 status to critical status; this critical status is indicated by the "X" through  
2 communication interface 20. After CS 12 has become unreachable, then MS  
3 14 takes over status polling for the collection station objects 16. Where there  
4 is an overlap between collection station objects 16 and management station  
5 objects 18, MS 14 will only poll a duplicate object one time. As previously  
6 discussed, if any duplicate objects are also polled by other collection  
7 station(s), not shown here, MS 14 as well as the other collection station(s) will  
8 both poll the duplicate object or node. This duplication of polling serves to  
9 maintain the polling configuration defined by the user.

10  
11 An overview of the methodology of the present invention is presented in  
12 **Figure 2**. First, as shown in Block 32, the initial configuration of the central  
13 control unit and one or more collection stations of the distributed computing  
14 environment is defined. During the initial configuration, the user specifies the  
15 polling configuration of the central control unit and the collection station(s). At  
16 Decision Block 34, the inquiry is whether a collection station of the distributed  
17 computing environment is not reachable by the central control unit. It is noted  
18 that since there may be a multitude of collection stations in the network, the  
19 network monitor program *netmon* will monitor each collection station in this  
20 manner. Thus, the flow 30 of **Figure 2** is performed for each collection station  
21 monitored by *netmon*.

22  
23 If the collection station at issue is reachable by the central control unit,

1 then *netmon* continues to monitor the collection station as indicated in the  
2 figure. If, however, the collection station has become unreachable, then a  
3 manual, or user initiated, failover of the collection station polling to the central  
4 control unit may be performed. As indicated by the dashed box of Block 36,  
5 the manual failover, as well as the manual release of Block 44, are optional.  
6 Normally, the manual failover and manual release of Blocks 36 and 44 would  
7 be not be performed in favor of the automatic failover described by Blocks 34  
8 and 42.

9  
10 Whether the collection station 12 has become unreachable is  
11 determined by the methodology illustrated in **Figure 3**. At Decision Block 52,  
12 the inquiry is whether the central control unit 14 has received an event over  
13 communication interface 20 from another process on the collection station  
14 which will determine if collection station 12 is unreachable. The event is  
15 normally sent after the collection station 12 has failed to respond to a  
16 predetermined number of polls, such as four polls, sent by a topology manager  
17 program of a topology database and the topology manager program therefore  
18 changes the status of the collection station to critical. If no such event is  
19 received by *netmon*, then the *netmon* program resident on the central control  
20 unit 14 simply continues to monitor the collection station 12 until such an event  
21 is received. If the central control unit 14 has received a collection station down  
22 event, then the flow continues to Decision Block 54. At Decision Block 54, the  
23 inquiry is whether a path to the collection station 12 is in the topology of the

1 central control unit 14. If it is, the inquiry at Block 56 is whether any  
2 intervening network objects are operational. If intervening network objects are  
3 down, then the station will not failover. If the station is not reachable because  
4 of a network device then the devices on the other side of the network device  
5 would also not be reachable. Failing over would be a waste of resources in  
6 this case.

7  
8 Referring back to **Figure 2**, once it is known that a monitored collection  
9 station is down, then failover polling status of the collection station's objects by  
10 the central control unit must occur. The failover and release of the failover,  
11 once the collection station is again able to handle status polling, may be either  
12 manual or automatic. Manual failover and release of the manual failover,  
13 shown in Blocks 36 and 44, are optional as indicated by the dashed boxes of  
14 these blocks. Manual failover and release indicate that the user must actively  
15 cause the failover and release operations, as the *netmon* program does for  
16 automatic failover and release. If manual failover and release are decided  
17 upon, then Blocks 36 and 44 replace Blocks 34 and 42 of Figure 2, the  
18 decision blocks that determine whether to continue polling.

19  
20 The first step of the automatic failover operation is to load the topology  
21 of the down collection station 12 as illustrated in Block 38. Loading the  
22 topology of the critical collection station is illustrated in more detail in the flow  
23 60 of **Figure 4**. Upon receiving the collection station down event, *netmon*

1 requests the topology for the collection station from the local topology  
2 datastore using any failover filter that has been specified by the user. As  
3 shown in Block 62, once *netmon* receives the collection station down event, it  
4 will request the list of the objects of the failed collection station that it is to poll  
5 from an application programming interface (API). This API will return the  
6 entire topology monitored by the critical collection station 12. *netmon* handles  
7 multiple loaded topologies in a linked list. The new loaded topology will not be  
8 merged with the current local loaded topology. In order to support multiple  
9 loaded topologies at the same time, the required data structure has a list of  
10 pointers to various topologies. As shown in **Figure 5**, the topologies for one or  
11 more critical collection stations may be loaded onto the central control unit.  
12 Referring to **Figure 6**, an implementation of Figure 5, the *key* of the data  
13 structure is the station id, a locally assigned id number, and *info* is a pointer to  
14 the topological information for a particular collection station. For example, as  
15 shown in the figure, key=0 for the local topology being monitored by the central  
16 control unit 14, key=171 for the topology of a first critical collection station and  
17 key=172 for the topology of a second critical collection station.

18  
19 The loaded topologies are stored, but *netmon* must determine which  
20 topologies to load at start-up. During a subsequent start-up of *netmon*,  
21 therefore, it will check the status for each of the collection stations of which it  
22 has knowledge. If any of the collection stations are down and the central  
23 control unit 14 has been configured to failover for the failed collection station(s)

1 in that circumstance, then the central control unit 14 will failover these stations.

2  
3 At Decision Block 64, the inquiry is whether the user has specified any  
4 filter through which the list of objects obtained as a result of Block 62 must be  
5 filtered. The user can specify a filter which allows the user to customize the  
6 objects that will be loaded when a failover occurs. The filter operates to filter  
7 out unspecified nodes and interfaces of the collection station objects; filters are  
8 applied on a per collection station basis. Since interfaces and the node to  
9 which they are attached are considered a unit, then if one interface or the  
10 nodes passes the filter than the entire node with all of its interfaces will pass  
11 the filter.

12  
13 When *netmon* requests the collection station topology to load via the  
14 API, this filter is applied to the objects of the collection station before the data  
15 is returned to *netmon*. It is more efficient to filter objects before the topology is  
16 provided to *netmon* than after it has been provided. Only the status polling  
17 functionality and not the full responsibilities of the collection station is taken  
18 over by the central control unit. The failover filter also gives the user the ability  
19 to determine how much extra work *netmon* will have to perform by controlling  
20 the collection station topology information that is provided to it.

21  
22 If such a failover filter has been specified, then the parameters of the  
23 filter must be applied at Block 66 to obtain the filtered list of objects. If no filter

1 has been specified, the flow continues directly to Block 68. At Block 68, the  
2 critical routes to the remaining objects to be polled must be calculated.  
3 *Netmon* calculates the critical route, if possible, for each of the nodes in the  
4 loaded topology in order to support the failover status polling.

5  
6 Referring back to **Figure 2**, *netmon* adds the filtered objects and the  
7 critical routings thereof to the status polling list of the central control unit in  
8 order to accomplish the failover polling of Block 40. The interfaces are added  
9 in such a manner as to avoid multiple polling of the same object, as discussed  
10 before. The status polling list is often referred to as a ping or IPXping list. The  
11 user, during the initial configuration of Block 32, determines whether the newly  
12 added objects will be polled at load time at Block 38 or at the time at which the  
13 critical collection station would have polled the objects. The user typically  
14 determines the frequency of polling; periodic polling, such as every five  
15 minutes, may be set by default. *Netmon* will send a station failover event to  
16 notify the user that status polling for the objects of the collection station 12  
17 have begun on the central control unit 14. Polling of the objects of the critical  
18 collection station by the central control unit 14 continues until the collection  
19 station 12 is back up and ready to resume polling of its own objects. This is  
20 illustrated by Decision Block 42. The placement of Block 42 after Block 40  
21 should not be construed to imply that an inquiry as to whether to continue  
22 polling is only performed after all of the objects have been polled. It is  
23 understood that the critical collection station is monitored continuously so that



1 whenever the collection station is again accessible the failover will be  
2 released, regardless of whether all objects have yet been polled or not. When  
3 the collection station 12 is back up again, then the failover is released – either  
4 manually at Block 44 or automatically at Block 46.

5  
6 Automatic release of the failover from the central control unit 14 back to  
7 a normal status collection station is initiated when *netmon* receives an event  
8 that indicates that the collection station 12 is reachable and able to perform  
9 polling operations of its own objects. To this end, the topology manager  
10 program of the topology database changes the status of a collection station to  
11 normal and triggers a collection station normal event to be sent. This  
12 collection station normal event is received by the *netmon* program of the  
13 central control unit 14. If the collection station 12 had failed over and central  
14 control unit 14 had taken over polling operations for it, then the central control  
15 unit 14 stops polling of the collection station objects, as indicated in Block 72  
16 of the flow 70 of **Figure 7**. Next, at Block 74 *netmon* unloads or deletes the  
17 loaded topology of the collection station that had failed and the objects that  
18 had been added to the status polling list of the central control unit 14. Once  
19 the topology of the collection station has been unloaded, a list of objects that  
20 have changed status since the collection station went critical must be obtained  
21 at Block 76. These objects typically would have changed as a result of status  
22 polling by the central control unit 14. As shown at Block 78, this list of  
23 changed collection station objects is used to synchronize the status on the

1 central control unit 14 and the collection station 12. Once the collection station  
2 regains normal status, the status of the objects according to the collection  
3 station 12 takes precedence over the status of the objects according to the  
4 central control unit 14. Therefore, the status of an object from the collection  
5 station 12 will take priority and overwrite the status for the same object stored  
6 in the central control unit 14 topology database. Once netmon has removed  
7 all references to the collection station, it sends a failover released message to  
8 the user to indicate that the central control unit 14 is no longer status polling  
9 for objects on the collection station 12.

10  
11  
12 The operation of the present invention is further demonstrated by the  
13 state diagram of **Figure 8**. STATION\_FAILOVER\_CHG\_EV (On) is an event  
14 that indicates that when a collection station goes down, the central control unit  
15 is to takeover status polling for the failed collection station.  
16 STATION\_FAILOVER\_CHG\_EV (Off) is an event that indicates that when a  
17 collection station goes down, the central control unit is not to takeover status  
18 polling. STATION\_FAILOVER\_FILTER\_CHG\_EV is an event that indicates  
19 that a filter is to filter the objects of the downed collection station prior to  
20 presenting the topology of the downed collected station to *netmon* of the  
21 central control unit; an existing topology can be reloaded using the filter.  
22 STATION\_DOWN\_EV is an event that indicates that a collection station has a  
23 critical status. STATION\_NORMAL\_EV is an event that indicates that a  
collection station has a normal status and is not down. As shown in the legend

1 of Figure 8, a dashed line indicates a transition that was triggered by a user  
2 action while a solid line indicates a transition that was triggered by a station  
3 event, such as a collection station or a central control unit event, and not by  
4 user intervention or action.

5  
6 The present invention uses software to perform network status polling,  
7 as opposed to the prior practice of accomplishing fault tolerance through  
8 hardware.

9  
10 While the invention has been particularly shown and described with  
11 reference to a preferred embodiment, it will be understood by those skilled in  
12 the art that various changes in form and detail may be made therein without  
13 departing from the spirit and scope of the invention.  
14